Synchronization of Multiple Videos
Supplementary Materials

Avihai Naaman
Ben Gurion University
avihaina@post.bgu.ac.il

Ron Shapira Weber
Ben Gurion University
ronsha@post.bgu.ac.il

Oren Freifeld
Ben Gurion University
oren.freifeld@gmail.com

GenAI Multiple Video Synchronization Dataset

Gen-MVS Dataset Details

Gen-MVS is a dataset of 82 AI-generated videos synthesized using prompts and images via `ChatGPT` and `KlingAI`, as described in the main paper. It contains 5 action classes with natural variation in visual style, motion speed, and subject identity (e.g., different instances of the same animal category, such as a bulldog and a German shepherd performing the same action). Each video is annotated with a *Start*, *End*, and one class-specific key event, supporting evaluation of multi-video synchronization (MVS), as summarized in Table 1.

Table 1. List of all key events in the Gen-MVS dataset. Each action has a *Start* event and *End* event in addition to the key event.

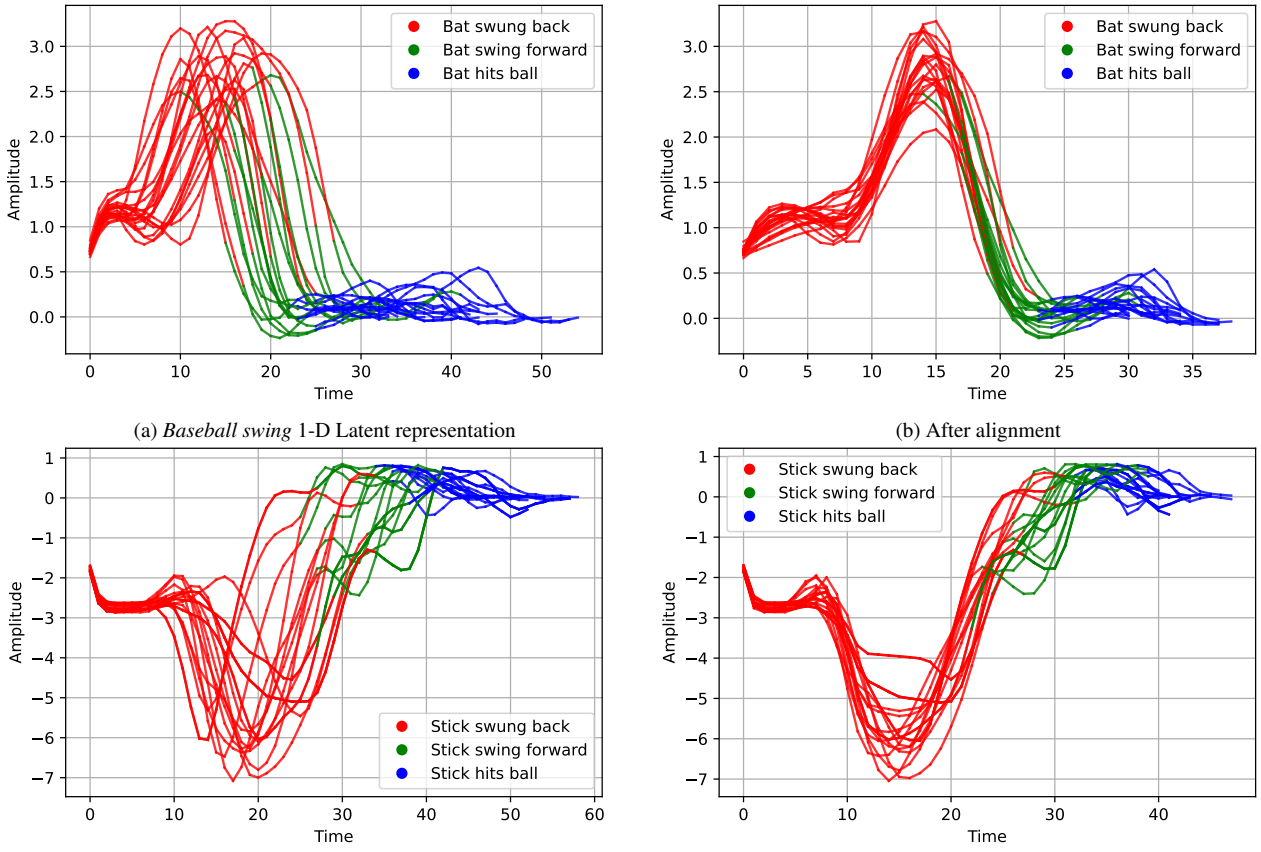
Action	**#phases**	**Key Event**	**Train set**	**Val set**
Bench-press	2	Bar fully down	9	5
Deadlift	2	Bar fully lifted	11	6
Dips	2	Elbows at 90°	12	6
Pullups	2	Chin above bar	11	5
Pushups	2	Head at floor	11	6

Annotation. All videos were manually filtered for visual and temporal quality, and annotated with per-video phase progression and key event frames. These annotations are used for both supervision and alignment evaluation.

Additional results

Joint alignment of video embeddings

The Temporal Prototype Learning (TPL) framework seeks to learn the joint alignment and temporal prototypes of video action sequences. It is achieved via the Diffeomorphic Multitasking Autoencoder (D-MTAE). The output of the encoder, Ψ_{encoder}, is a 1-D representation of the multi-channel inputs, which are then jointly-aligned by a Diffeomorphic Temporal Alignment Net (DTAN) [6], Ψ_{DTAN}. Figure 1 presents the alignment results on the *Baseball swing* and *Golf swing* actions.



(a) *Baseball swing* 1-D Latent representation

(b) After alignment

(c) *golf swing* 1-D Latent representation

(d) After alignment

Figure 1. Joint alignment of the one-dimensional representation for 20 randomly sampled videos of the *Baseball swing* (top) and *golf swing* (bottom) action colored by the phase labels, before (left) and after alignment (right).

Multiple video synchronization

We provide qualitative results demonstrating the effectiveness of our Temporal Prototype Learning (TPL) for synchronizing multiple unsynchronized videos of the same action. TPL leverages learned temporal embeddings to align sequences with high temporal fidelity, even under challenging conditions such as viewpoint variation and subtle temporal misalignments. As shown in Figure 2, Figure 3, Figure 4, and Figure 5, TPL accurately estimates temporal offsets and brings semantically corresponding frames into alignment across multiple video sources.

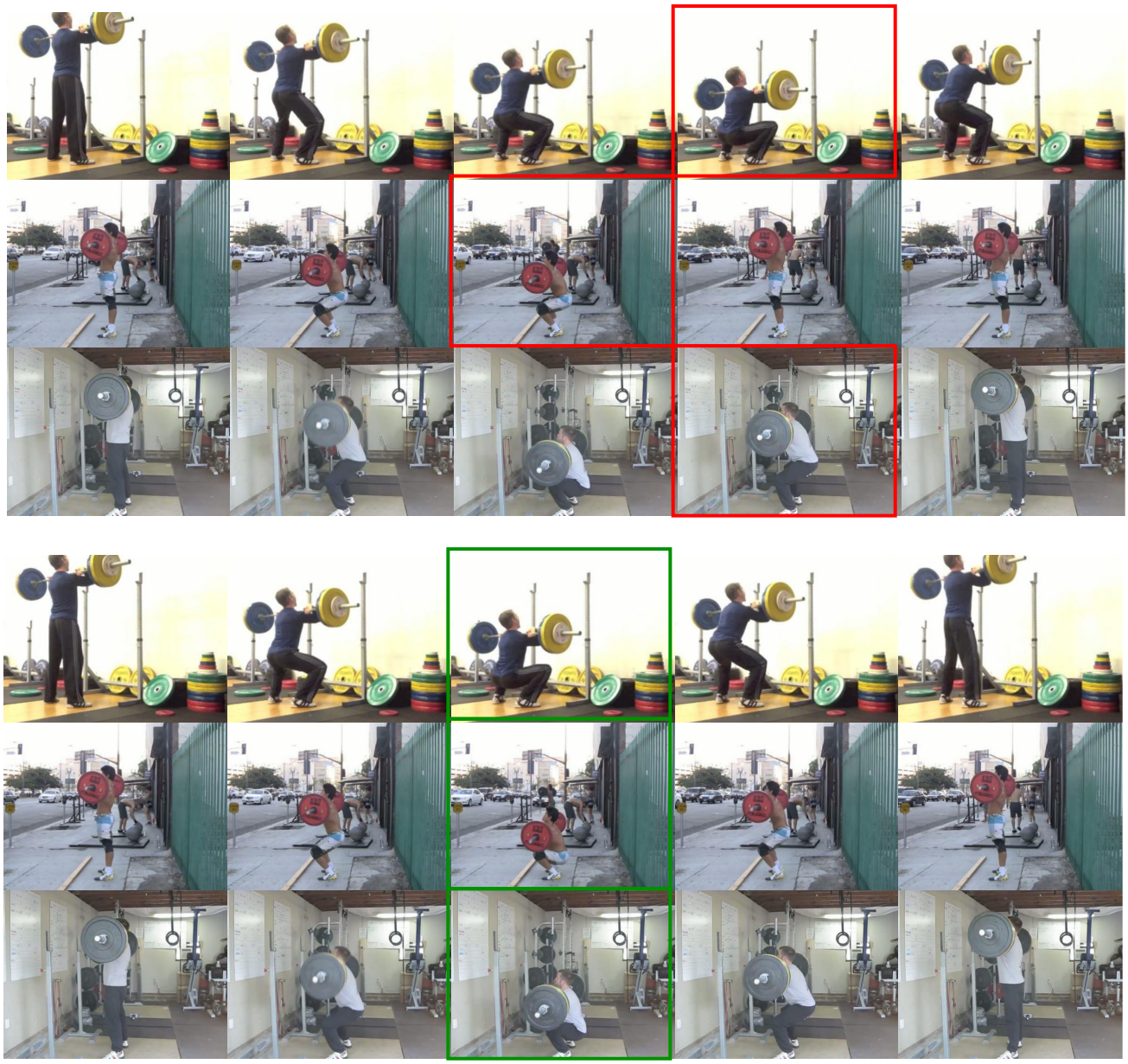


Figure 2. Examples from Penn Action dataset (squat action), showing five equally spaced frames before and after synchronization. We highlight mismatches in the original videos in red, and TPL matching in green. Alignment via TPL successfully synchronizes the key phases of the squat action.

Figure 3. Examples from Penn Action dataset (baseball pitch action), showing five equally spaced frames before and after synchronization. We highlight mismatches in the original videos in red, and TPL matching in green. Alignment via TPL successfully synchronizes the key phases of the baseball pitch action.

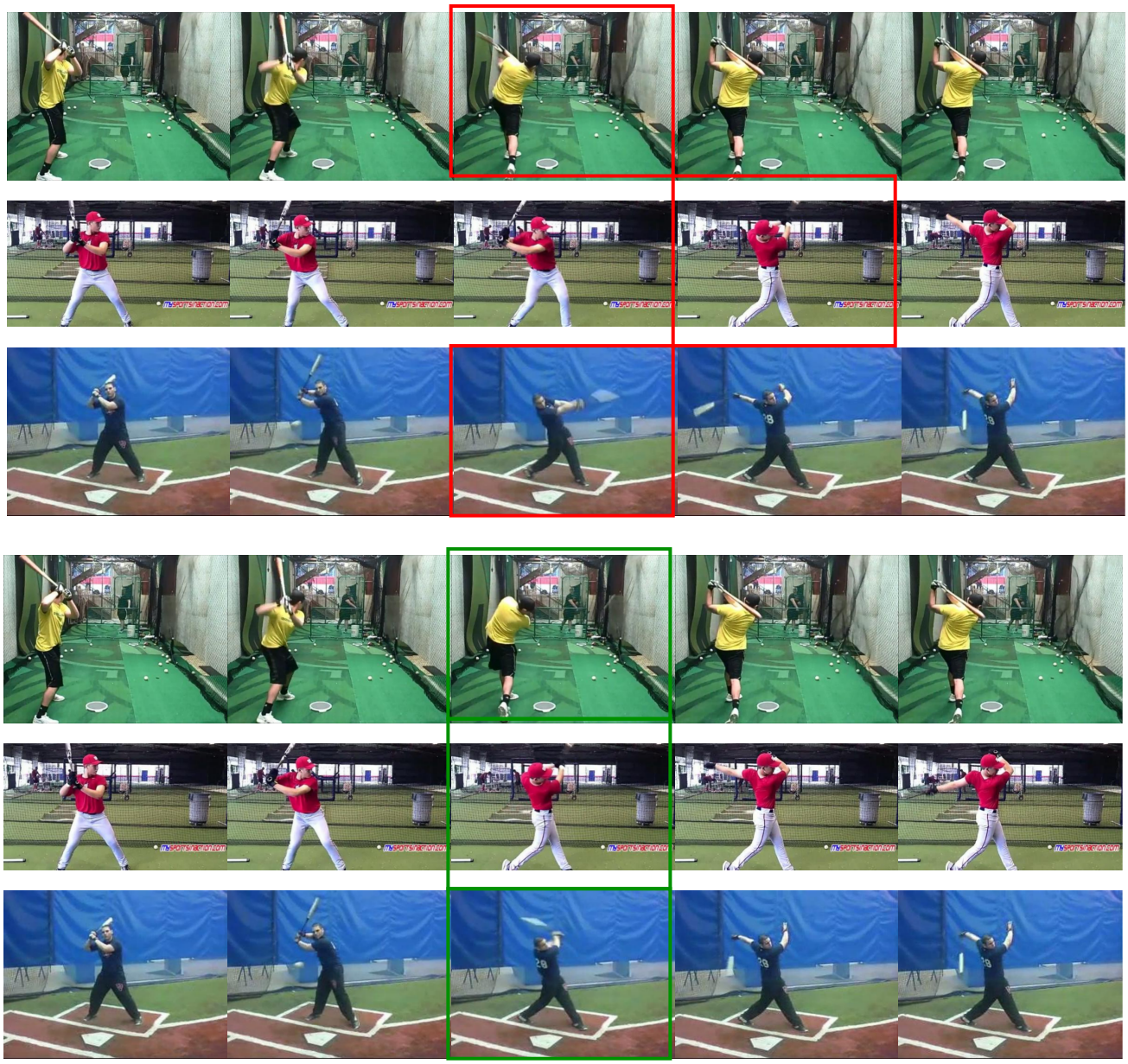


Figure 4. Examples from Penn Action dataset (baseball swing action), showing five equally spaced frames before and after synchronization. We highlight mismatches in the original videos in red, and TPL matching in green. Alignment via TPL successfully synchronizes the key phases of the baseball swing action.

Figure 5. Examples from Penn Action dataset (tennis forehand action), showing five equally spaced frames before and after synchronization. We highlight mismatches in the original videos in red, and TPL matching in green. Alignment via TPL successfully synchronizes the key phases of the tennis forehand action.

Implementation Details

Diffeomorphic Multitasking Autoencoder (D-MTAE) modules architecture are presented in Table 2 and Table 3.

Table 2. D-MTAE modules architecture.

Operations	Output Size	Parameters
Encoder network – $\psi_{\text{encoder}}(\cdot)$		
Conv1d	128	[input_channels, 128, 3, padding=1]
BatchNorm1d	128	—
GELU	128	—
Conv1d	64	[128, 64, 3, padding=1]
BatchNorm1d	64	—
GELU	64	—
Conv1d	32	[64, 32, 3, padding=1]
BatchNorm1d	32	—
GELU	32	—
Conv1d	16	[32, 16, 3, padding=1]
BatchNorm1d	16	—
GELU	16	—
Conv1d	1	[16, 1, 3, padding=1]
Decoder network – $\psi_{\text{decoder}}(\cdot)$		
ConvTranspose1d	16	[1, 16, 3, padding=1]
BatchNorm1d	16	—
GELU	16	—
ConvTranspose1d	32	[16, 32, 3, padding=1]
BatchNorm1d	32	—
GELU	32	—
ConvTranspose1d	64	[32, 64, 3, padding=1]
BatchNorm1d	64	—
GELU	64	—
ConvTranspose1d	128	[64, 128, 3, padding=1]
BatchNorm1d	128	—
GELU	128	—
ConvTranspose1d	input_channels	[128, input_channels, 3, padding=1]

Table 3. Joint alignment network - $\psi_{\text{Align}}(\cdot)$

Operations	Output Size	Parameters
Inception Block		
Bottleneck Conv	32	$[c, 1, 32]$
Conv	32	[32, 39, 32]
Conv	32	[32, 19, 32]
Conv	32	[32, 9, 32]
Max Pooling	c	—
Conv	32	$[c, 1, 32]$
Concatenation	128	—
Batch Norm	128	—
ReLU	128	—
Shortcut		
Conv	128	$[c, 1, 128]$
Batch Norm	128	—
Batch Norm	128	—
Addition	128	—
ReLU	128	—
Alignment Head		
GAP	128	—
Flatten	128	—
Linear Projection	$\dim(\boldsymbol{\theta})$	$[128, \dim(\boldsymbol{\theta})]$

Training details

For $\psi_{\text{Align}}(\cdot)$, we set the number of cells in the partition of the velocity field to $N_p = 16$. We enforce the boundary condition ($v^{\boldsymbol{\theta}}[0] = v^{\boldsymbol{\theta}}[16] = 0$) and thus $\dim(\boldsymbol{\theta}) = 15$. We use the InceptionTime backbone for the localization net [3] and use the implementation from `tsai` [5]. As for the training procedure, we set the batch size to 64 with a learning rate of 10^{-4}. We jointly train the D-MTAE for all classes for 300 epochs using the AdamW optimizer [4] with a weight decay of 10^{-4}. We use a 4090 RTX graphic card for the training of all models.

VAE Variant for OpenCLIP and DINO Features

For experiments involving pretrained embeddings from OpenCLIP [2] and DINO [1], we modify the D-MTAE architecture by replacing the standard autoencoder with a **Variational Autoencoder (VAE)**.

Latent Sampling. The encoder Ψ_{encoder} now produces a latent distribution per timestep, returning per-frame means $\mu_i \in \mathbb{R}^{L_i}$ and log-variances $\log \sigma_i^2 \in \mathbb{R}^{L_i}$. The latent trajectory $Z_i \in \mathbb{R}^{L_i}$ is sampled as:

$$Z_i[t] = \mu_i[t] + \epsilon_i[t] \cdot \sigma_i[t], \quad \epsilon_i[t] \sim \mathcal{N}(0, 1) \tag{1}$$

Reconstruction Loss. After alignment, the decoder reconstructs $\widetilde{U}_i = \Psi_{\text{decoder}}(\widetilde{Z}_i)$ as in the main paper. To handle potential missing or invalid inputs, we use a **masked reconstruction loss**:

$$\mathcal{L}_{\text{rec}} = \frac{1}{N} \sum_{i=1}^{N} \frac{\left\| M_i \odot (U_i - \widetilde{U}_i \circ T^{-\boldsymbol{\theta}_i}) \right\|^2}{\sum M_i} \tag{2}$$

where $M_i \in \{0,1\}^{C \times L_i}$ is a binary mask and $\odot$ denotes element-wise multiplication.

KL Divergence. To ensure the latent distribution is centered and standardized over time, we apply a **masked KL divergence loss**:

$$\mathcal{L}_{\text{KL}} = \frac{1}{N} \sum_{i=1}^{N} \frac{\sum_{t=1}^{L_i} m_i^{(z)}[t] \cdot \left(-\frac{1}{2} \left(1 + \log \sigma_i[t]^2 - \mu_i[t]^2 - \sigma_i[t]^2 \right) \right)}{\sum_t m_i^{(z)}[t]} \tag{3}$$

where $m_i^{(z)}[t] \in \{0,1\}$ is a reduced (per-frame) validity mask.

Temporal Smoothness. To further regularize the temporal latent trajectories, we penalize sudden changes in Z_i via a **smoothness loss**:

$$\mathcal{L}_{\text{smooth}} = \frac{1}{N} \sum_{i=1}^{N} \frac{\sum_{t=1}^{L_i - 1} m_i^{(z)}[t] \cdot m_i^{(z)}[t+1] \cdot (Z_i[t+1] - Z_i[t])^2}{\sum_t m_i^{(z)}[t] \cdot m_i^{(z)}[t+1]} \tag{4}$$

Trajectory Variance Loss. To encourage coherent temporal dynamics across sequences in a batch, we introduce a trajectory variance loss. For each sequence, we extract $Z_i^{\text{sub}} \in \mathbb{R}^K$ by uniformly sampling K valid timesteps from the latent trajectory Z_i. The loss penalizes deviation from the batch-wise mean trajectory:

$$\mathcal{L}_{\text{traj-var}} = \frac{1}{N} \sum_{i=1}^{N} \left\| Z_i^{\text{sub}} - \overline{Z}^{\text{sub}} \right\|^2, \tag{5}$$

where $\overline{Z}^{\text{sub}} = \frac{1}{N} \sum_{i=1}^{N} Z_i^{\text{sub}}$. This regularization encourages latent trajectories to evolve with similar temporal structure across samples, without enforcing identity or similarity in content.

Final Objective. The total training loss used for OpenCLIP/DINO features becomes:

$$\mathcal{L}_{\text{TPL-VAE}} = \lambda_t \mathcal{L}_{\text{ICAE}} + \mathcal{L}_{\text{rec}} + \beta \cdot \mathcal{L}_{\text{KL}} + \gamma \cdot \mathcal{L}_{\text{smooth}} + \alpha \cdot \mathcal{L}_{\text{traj-var}} \tag{6}$$

where λ_t is the annealed ICAE weight, and β, γ, α are fixed hyperparameters.

Notes. - When using this variant, only the encoder and decoder are changed; the alignment module Ψ_{Align} and CPAB transformations remain as described in the main paper. - This change is only applied to experiments where input features are obtained from pretrained OpenCLIP or DINO models.

References

[1] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021. 8

[2] Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. Reproducible scaling laws for contrastive language-image learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2818–2829, 2023. 8

[3] Hassan Ismail Fawaz, Benjamin Lucas, Germain Forestier, Charlotte Pelletier, Daniel F Schmidt, Jonathan Weber, Geoffrey I Webb, Lhassane Idoumghar, Pierre-Alain Muller, and François Petitjean. Inceptiontime: Finding alexnet for time series classification. *Data Mining and Knowledge Discovery*, 2020. 8

[4] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. 8

[5] Ignacio Oguiza. tsai - a state-of-the-art deep learning library for time series and sequential data. Github, 2022. 8

[6] Ron Shapira Weber, Matan Eyal, Nicki Skafte Detlefsen, Oren Shriki, and Oren Freifeld. Diffeomorphic temporal alignment nets. In *NeurIPS*, 2019. 3